

# Report for ASE - Homewok 1

Crea Giuseppe, #501922

## Requirements:

Complete the parties.py view within the given files.

## Implementation:

The various decorations were missing which methods were allowed, that was handled according to the provided documentation.

The various methods were missing their bodies, but as they are quite short a detailed explanation seems superfluous. Special care was taken in handling all exceptions thrown by the party.py class. To speed up and be consistent with the implementation, pre-existing functions were used whenever possible. No design liberties were taken. Return messages were tailored to the API. What was not specified in the documentation was recovered from the test file. When neither the documentation nor the test file specified a message (cases in which only a code was requested), a fitting one was implemented, for human readability.

Some methods (single\_party for instance) don't make use of the pre-defined return variable "result", but this doesn't change anything at an API logic level.

Repo link: [https://github.com/giuseppe-crea/ASE\\_Assignment\\_1](https://github.com/giuseppe-crea/ASE_Assignment_1)

## Screenshots

Operations sorted by their order within the test file.

```
----- coverage: platform win32, python 3.9.0-final-0 -----
Name                               Stats    Miss Branch BrPart  Cover   Missing
-----
bedrock_a_party\classes\party.py    64      8    10      2    86%  17, 31, 39-40, 99, 102, 107, 110
bedrock_a_party\views\parties.py    73     11    22      5    81%  25->29, 49-53, 64->67, 85-86, 93-94, 112-113, 135
-----
TOTAL                               188     19    32      7    87%

6 files skipped due to complete coverage.
Coverage HTML written to dir htmlcov

***** Fail: Required test coverage of 98% not reached. Total coverage: 87.27% *****
***** 2 passed, 17 warnings in 1.67s *****
PS I:\Documents\OneDrive Unipi\OneDrive - University of Pisa\Corsi di Laurea\Cybersecurity LM-66\Secondo Anno\Primo Semestre\Advanced Software Engineering\Python\skeleton\ase-assignment1-master> |
```

### 1- Test File Success

## 1<sup>st</sup> Test Case

The screenshot displays the Postman interface for a REST client. The workspace is named "Homework 1" and contains a collection of requests. The selected request is a POST to "http://127.0.0.1:5000/parties". The body is set to JSON and contains the following data:

```
{  "guests": [    "Giacomo",    "Francesco",    "Federico"]  }
```

The response is a 200 OK status with a response time of 20 ms and a body size of 164 B. The response body is shown in the "Body" tab, formatted as JSON:

```
{  "party_number": 0}
```

The terminal window at the bottom shows the Flask application running on http://127.0.0.1:5000/. The output includes the following lines:

```
FLASK_DEBUG = 0
In folder I:\Documents\OneDrive Unipi\OneDrive - University of Pisa\Corsi di Laurea\Cybersecurity LM-66\Secondo Anno\Primo Semestre\Advanced Soft
"I:\Documents\OneDrive Unipi\OneDrive - University of Pisa\Corsi di Laurea\Cybersecurity LM-66\Secondo Anno\Primo Semestre\Advanced Software Eng
* Serving Flask app "bedrock_a_party"
* Environment: development
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [08/Oct/2021 20:49:27] "POST /parties HTTP/1.1" 200 -
```

## 2- POST a new party in /parties

The screenshot displays the Postman application interface. On the left, the 'My Workspace' sidebar shows a collection named 'Homework 1' with several API requests. The main panel shows a 'POST' request to 'http://127.0.0.1:5000/parties'. The request body is a JSON object with a 'guests' array containing three names: 'Giacomo', 'Francesco', and 'Federico'. The response is a JSON object with 'party\_number': 2. The status bar at the bottom indicates a 200 OK response with a 7 ms latency and 164 B of data. A terminal window at the bottom shows the command 'python3 -m http.server 5000' and the output of the POST requests.

```
POST http://127.0.0.1:5000/parties
{
  "guests": [
    "Giacomo",
    "Francesco",
    "Federico"
  ]
}
```

```
200 OK 7 ms 164 B
{"party_number": 2}
```

```
I:\Documents\OneDrive Unipi\OneDrive - University of Pisa\Corsi di Laurea\Cybersecurity LM-66\Secondo Anno\Primo Semestre\Advanced Software Eng
* Serving Flask app "bedrock_a_party"
* Environment: development
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [08/Oct/2021 20:49:27] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:49:58] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:49:59] "POST /parties HTTP/1.1" 200 -
```

3- POST two additional copies of that party

Postman

File Edit View Help

Home Workspaces Reports Explore

Search Postman

My Workspace

New Import

Homework 1

- POST POST new party to parties
- GET GET parties
- GET GET party 0
- POST POST food item from valid user ...
- GET GET foodlist party 2
- GET GET loaded parties
- POST POST food item from invalid us...
- POST POST empty party
- DELETE DELETE valid food item Party 2 ...
- DELETE DELETE invalid food item Party 2
- DELETE DELETE party 2

Homework 1 / GET loaded parties

GET http://127.0.0.1:5000/parties/loaded ...

Params Auth Headers (6) Body Pre-req. Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (4) Test Results

200 OK 7 ms 166 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "loaded_parties": 3
3 }
```

Find and Replace Console

Bootcamp Runner Trash

```
* Serving Flask app "bedrock_a_party"
* Environment: development
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [08/Oct/2021 20:49:27] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:49:58] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:49:59] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:50:28] "GET /parties/loaded HTTP/1.1" 200 -
```

3- GET /parties/loaded

The screenshot displays the Postman application interface. On the left sidebar, under 'My Workspace', there is a collection named 'Homework 1' containing several API endpoints. The main panel shows a 'GET' request to 'http://127.0.0.1:5000/parties'. The response is a JSON array of party objects, displayed in a 'Pretty' format. The bottom console shows the execution of several HTTP requests and their 200 OK responses.

**Request Details:**

- Method: GET
- URL: http://127.0.0.1:5000/parties

**Response (Pretty):**

```
[{"loaded_parties": [{"foodlist": [], "guests": ["Giacomo", "Francesco", "Federico"], "id": 0}, {"foodlist": [], "guests": ["Giacomo", "Francesco", "Federico"], "id": 1}, {"foodlist": [], "guests": ["Giacomo", "Francesco", "Federico"], "id": 2}]]
```

**Console Log:**

```
* Environment: development
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [08/Oct/2021 20:49:27] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:49:58] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:49:59] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:50:28] "GET /parties/loaded HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:50:50] "GET /parties HTTP/1.1" 200 -
```

4- GET /parties, return json shrunk for readability

The screenshot displays the Postman application window. The top bar includes the Postman logo, menu items (File, Edit, View, Help), and navigation tabs (Home, Workspaces, Reports, Explore). A search bar and utility buttons (Invite, Upgrade) are also present.

The main workspace is titled "My Workspace" and shows a collection of API requests under "Homework 1". The selected request is "POST empty party". The request details are as follows:

- Method:** POST
- URL:** http://127.0.0.1:5000/parties ...
- Body:** raw JSON, `{ "guests": [] }`

The response section shows a 400 BAD REQUEST status with a response time of 9 ms and a size of 299 B. The response body is displayed in Pretty format:

```
{
  "code": 400,
  "description": "'You cannot create a party without guests'",
  "message": "400 Bad Request: 'You cannot create a party without guests'"
}
```

The bottom console window shows the following log output:

```
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [08/Oct/2021 20:49:27] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:49:58] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:49:59] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:50:28] "GET /parties/loaded HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:50:50] "GET /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:51:22] "POST /parties HTTP/1.1" 400 -
```

5- POST /parties with empty party

## 2<sup>nd</sup> Test

The screenshot displays the Postman application interface. The top bar includes the Postman logo, menu options (File, Edit, View, Help), and a search bar. The main workspace is divided into three sections: a sidebar on the left for collections and environments, a central area for request details, and a bottom console for logs.

The sidebar shows a collection named "Homework 1" with several requests. The selected request is "GET party 2". The main workspace shows the details of this request, including the method (GET), the URL (http://127.0.0.1:5000/party/2), and the response body. The response body is a JSON object with the following structure:

```
{  "foodlist": [],  "guests": [    "Giacomo",    "Francesco",    "Federico"  ],  "id": 2}
```

The bottom console shows the execution logs, indicating that the request was successful (200 OK) and the response was received.

6- GET /party/2

Postman

File Edit View Help

Home Workspaces Reports Explore

Search Postman

My Workspace

New Import

Homework 1

POST POST new party to parties

GET GET parties

GET GET party 2

GET GET party 100

POST POST food item from valid user ...

GET GET foodlist party 2

GET GET loaded parties

POST POST food item from invalid us...

POST POST empty party

DEL DELETE valid food item Party 2 ...

DEL DELETE invalid food item Party 2

DEL DELETE party 2

Homework 1 / GET party 100

GET http://127.0.0.1:5000/party/100

Send

Params Auth Headers (6) Body Pre-req. Tests Settings

Query Params

KEY	VALUE	DESCRIPTION	Bulk Edit
Key	Value	Description	

Body Cookies Headers (4) Test Results

404 NOT FOUND 12 ms 451 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "code": 404,
3   "description": "The requested URL was not found on the server. If you entered the URL
4   manually please check your spelling and try again.",
5   "message": "404 Not Found: The requested URL was not found on the server. If you entered the
6   URL manually please check your spelling and try again."
}
```

Find and Replace Console

Bootcamp Runner Trash

```
127.0.0.1 - - [08/Oct/2021 20:49:27] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:49:58] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:49:59] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:50:28] "GET /parties/loaded HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:50:50] "GET /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:51:22] "POST /parties HTTP/1.1" 400 -
127.0.0.1 - - [08/Oct/2021 20:51:52] "GET /party/2 HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:52:38] "GET /party/100 HTTP/1.1" 404 -
```

7- GET /party/100



Postman interface showing a workspace named "Homework 1" with a collection of API requests. The selected request is "POST food item from valid user ..." (highlighted in yellow). The request details show a POST method to the URL `http://127.0.0.1:5000/party/2/foodlist/Francesco/pizza...`. The request body is a JSON object:

```
{  "food": "pizza",  "user": "Francesco"}
```

The response status is 200 OK, 8 ms, 181 B. The response body is displayed in the "Body" tab, showing the JSON object above.

The console at the bottom shows the following log entries:

```
127.0.0.1 - - [08/Oct/2021 20:49:58] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:49:59] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:50:28] "GET /parties/loaded HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:50:50] "GET /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:51:22] "POST /parties HTTP/1.1" 400 -
127.0.0.1 - - [08/Oct/2021 20:51:52] "GET /party/2 HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:52:38] "GET /party/100 HTTP/1.1" 404 -
127.0.0.1 - - [08/Oct/2021 20:53:07] "POST /party/2/foodlist/Francesco/pizza HTTP/1.1" 200 -
```

8- POST a new food item in existing party from guest of that party

The screenshot displays the Postman application interface. The top bar includes the Postman logo, menu items (File, Edit, View, Help), and navigation tabs (Home, Workspaces, Reports, Explore). A search bar and utility buttons (Invite, Upgrade) are also present.

The left sidebar shows the 'My Workspace' section with a list of collections. The 'Homework 1' collection is expanded, showing several API requests. The selected request is 'POST food item from invalid user party 2'.

The main panel shows the details of the selected request. The method is 'POST' and the URL is 'http://127.0.0.1:5000/party/2/foodlist/Marco/pizza'. The request body is empty, indicated by the message 'This request does not have a body'.

The response is displayed in the 'Body' tab, showing a 401 Unauthorized status. The response body is a JSON object:

```
{
  "code": 401,
  "description": "'Marco is not invited to this party'",
  "message": "401 Unauthorized: 'Marco is not invited to this party'"
}
```

The bottom panel shows the console output, displaying the raw HTTP request and response logs. The logs show the following sequence of requests and responses:

```
127.0.0.1 - - [08/Oct/2021 20:49:59] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:50:28] "GET /parties/loaded HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:50:50] "GET /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:51:22] "POST /parties HTTP/1.1" 400 -
127.0.0.1 - - [08/Oct/2021 20:51:52] "GET /party/2 HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:52:38] "GET /party/100 HTTP/1.1" 404 -
127.0.0.1 - - [08/Oct/2021 20:53:07] "POST /party/2/foodlist/Francesco/pizza HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:53:40] "POST /party/2/foodlist/Marco/pizza HTTP/1.1" 401 -
```

9- POST a new food item to existing party from someone who is NOT a guest of that party

Postman

File Edit View Help

Home Workspaces Reports Explore

Search Postman

My Workspace

New Import

Homework 1 / GET foodlist party 2

GET http://127.0.0.1:5000/party/2/foodlist ...

Params Auth Headers (6) Body Pre-req. Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (4) Test Results

200 OK 7 ms 196 B Save Response

Pretty Raw Preview Visualize JSON

```
1 2 3 4 5 6 7 8
{
  "foodlist": [
    {
      "food": "pizza",
      "user": "Francesco"
    }
  ]
}
```

Find and Replace Console

Bootcamp Runner Trash

```
127.0.0.1 - - [08/Oct/2021 20:50:28] "GET /parties/loaded HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:50:50] "GET /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:51:22] "POST /parties HTTP/1.1" 400 -
127.0.0.1 - - [08/Oct/2021 20:51:52] "GET /party/2 HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:52:38] "GET /party/100 HTTP/1.1" 404 -
127.0.0.1 - - [08/Oct/2021 20:53:07] "POST /party/2/foodlist/Francesco/pizza HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:53:40] "POST /party/2/foodlist/Marco/pizza HTTP/1.1" 401 -
127.0.0.1 - - [08/Oct/2021 20:53:57] "GET /party/2/foodlist HTTP/1.1" 200 -
```

10- GET party/2/foodlist

Postman

File Edit View Help

Home Workspaces Reports Explore

Search Postman

My Workspace

New Import

Homework 1

- POST POST new party to parties
- GET GET parties
- GET GET party 2
- GET GET party 100
- POST POST food item from valid user ...
- GET GET foodlist party 2
- GET GET loaded parties
- POST POST food item from invalid us...
- POST POST empty party
- DEL DELETE valid food item Party 2 ...
- DEL DELETE invalid food item Party 2
- DEL DELETE party 2

Homework 1 / DELETE valid food item Party 2 Copy

DELETE http://127.0.0.1:5000/party/2/foodlist/Francesco/pizza ...

Params Auth Headers (6) Body Pre-req. Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (4) Test Results

200 OK 8 ms 169 B Save Response

Pretty Raw Preview Visualize JSON

```
1  {
2    "msg": "Food deleted!"
3  }
```

Find and Replace Console

Bootcamp Runner Trash

```
127.0.0.1 - - [08/Oct/2021 20:50:50] "GET /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:51:22] "POST /parties HTTP/1.1" 400 -
127.0.0.1 - - [08/Oct/2021 20:51:52] "GET /party/2 HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:52:38] "GET /party/100 HTTP/1.1" 404 -
127.0.0.1 - - [08/Oct/2021 20:53:07] "POST /party/2/foodlist/Francesco/pizza HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:53:40] "POST /party/2/foodlist/Marco/pizza HTTP/1.1" 401 -
127.0.0.1 - - [08/Oct/2021 20:53:57] "GET /party/2/foodlist HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:54:16] "DELETE /party/2/foodlist/Francesco/pizza HTTP/1.1" 200 -
```

11- DELETE valid food item from foodlist

## Outside of Tests:

The screenshot shows the Postman interface. On the left, a collection named 'Homework 1' is expanded, showing a list of requests. The selected request is 'DELETE party 2'. The main panel shows the details of this request: the method is 'DELETE', the URL is 'http://127.0.0.1:5000/party/2...', and the body is empty. The 'Send' button is visible. Below the request details, the response is shown in the 'Body' tab, which is formatted as JSON: `{ "msg": "Party deleted!" }`. The status bar at the bottom indicates a 200 OK response with a 14 ms response time and 170 B of data. At the very bottom, a terminal window shows the output of a Flask application, including the message 'Serving Flask app "bedrock\_a\_party"' and several log entries for POST and DELETE requests.

12- Delete an existing Party (lines 52-56 of parties.py)

**Postman**

File Edit View Help

Home Workspaces Reports Explore

Search Postman

My Workspace

New Import

Homework 1

- POST POST new party to parties
- GET GET parties
- GET GET party 2
- GET GET party 100
- POST POST food item from valid user ...
- GET GET foodlist party 2
- GET GET loaded parties
- POST POST food item from invalid us...
- POST POST empty party
- DEL DELETE valid food item Party 2 ...
- DEL DELETE invalid food item Party 2
- DEL DELETE party 2

Homework 1 / DELETE valid food item Party 2 Copy

DELETE http://127.0.0.1:5000/party/2/foodlist/Francesco/pizza ...

Params Auth Headers (6) Body Pre-req. Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	Bulk Edit
Key	Value	Description	

Body Cookies Headers (4) Test Results 400 BAD REQUEST 18 ms 333 B Save Response

Pretty Raw Preview Visualize JSON

```

1 {
2   "code": 400,
3   "description": "'user Francesco has not added pizza to this party foodlist'",
4   "message": "400 Bad Request: 'user Francesco has not added pizza to this party foodlist'"
5 }

```

Find and Replace Console

```

127.0.0.1 - - [08/Oct/2021 20:51:22] "POST /parties HTTP/1.1" 400 -
127.0.0.1 - - [08/Oct/2021 20:51:52] "GET /party/2 HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:52:38] "GET /party/100 HTTP/1.1" 404 -
127.0.0.1 - - [08/Oct/2021 20:53:07] "POST /party/2/foodlist/Francesco/pizza HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:53:40] "POST /party/2/foodlist/Marco/pizza HTTP/1.1" 401 -
127.0.0.1 - - [08/Oct/2021 20:53:57] "GET /party/2/foodlist HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:54:16] "DELETE /party/2/foodlist/Francesco/pizza HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 20:54:35] "DELETE /party/2/foodlist/Francesco/pizza HTTP/1.1" 400 -

```

13- Delete a non-existent food item from a party, through a valid guest (lines 39-40 of party.py and 96-97 of parties.py)

Postman interface showing a failed POST request. The request is to `http://127.0.0.1:5000/party/2/foodlist/Francesco/pizza...` and returns a 400 BAD REQUEST. The response body is:

```

{
  "code": 400,
  "description": "'Francesco already committed to bring pizza'",
  "message": "400 Bad Request: 'Francesco already committed to bring pizza'"
}

```

The console at the bottom shows the sequence of requests, with the last one failing with a 400 status:

```

* Environment: development
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [08/Oct/2021 21:01:01] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 21:01:01] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 21:01:02] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 21:01:07] "POST /party/2/foodlist/Francesco/pizza HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 21:01:19] "POST /party/2/foodlist/Francesco/pizza HTTP/1.1" 400 -

```

14- One user trying to insert the same food twice (lines 30-31 of `party.py` and 88-89 of `parties.py`)

Postman

File Edit View Help

Home Workspaces Reports Explore

Search Postman

My Workspace

New Import

Homework 1 / GET party 2

GET http://127.0.0.1:5000/party/2

Params Auth Headers (6) Body Pre-req. Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (4) Test Results

410 GONE 6 ms 551 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "code": 410,
3   "description": "The requested URL is no longer available on this server and there is no
4   forwarding address. If you followed a link from a foreign page, please contact the author
5   of this page.",
6   "message": "410 Gone: The requested URL is no longer available on this server and there is no
7   forwarding address. If you followed a link from a foreign page, please contact the author
8   of this page."
9 }
```

Find and Replace Console

Environment: development

Debug mode: off

Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)

```
127.0.0.1 - - [08/Oct/2021 21:05:19] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 21:05:20] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 21:05:20] "POST /parties HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 21:05:22] "DELETE /party/2 HTTP/1.1" 200 -
127.0.0.1 - - [08/Oct/2021 21:06:08] "GET /party/2 HTTP/1.1" 410 -
```

15- Trying to access a deleted party (line 138, parties.py)